Collaborative infrastructure

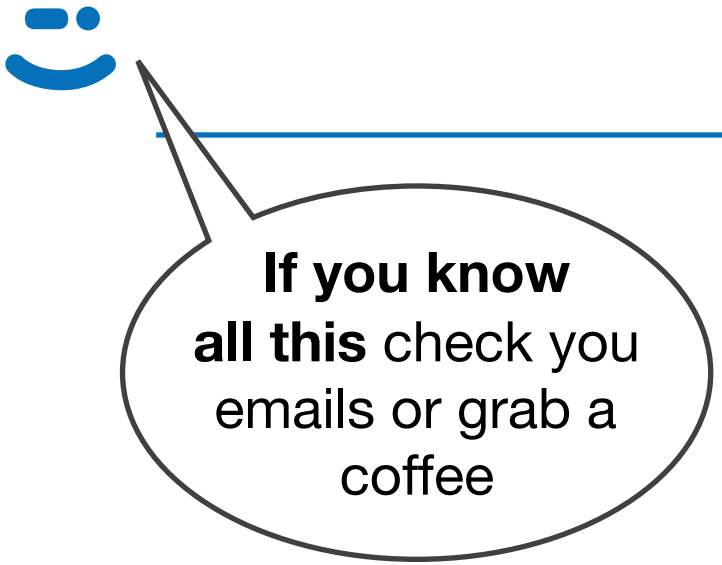
Tommaso Vinci

SMILEI training workshop
November 6-7, 2017
Maison de la Simulation

collaborative tools

conflicts	:	Git
build	:	Makefile
documentation	:	Sphinx/Travis
validation	:	Gitlab-CI

collaborative tools



conflicts	:	Git
build	:	Makefile
documentation	:	Sphinx/Travis
validation	:	Gitlab-CI

some numbers

17 developers

Julien Derouillat	1058
Fred Perez	878
Tommaso Vinci	655
Arnaud Beck	635
Mickael Grech	278
Mathieu Lobet	275
Marco Chiaramello	72
Marie Fle	45
Haithem Kallala	30
Charles Ruyer	20
Mohamed Gaalich	10
Andrea Sgattoni	7
Jeremie Gaidamour	6
Anna Grassi	9
Jérémy Dargent	4
Nicolas Aunai	4
Anton Golovanov	2
Total commits	**3988**

some numbers

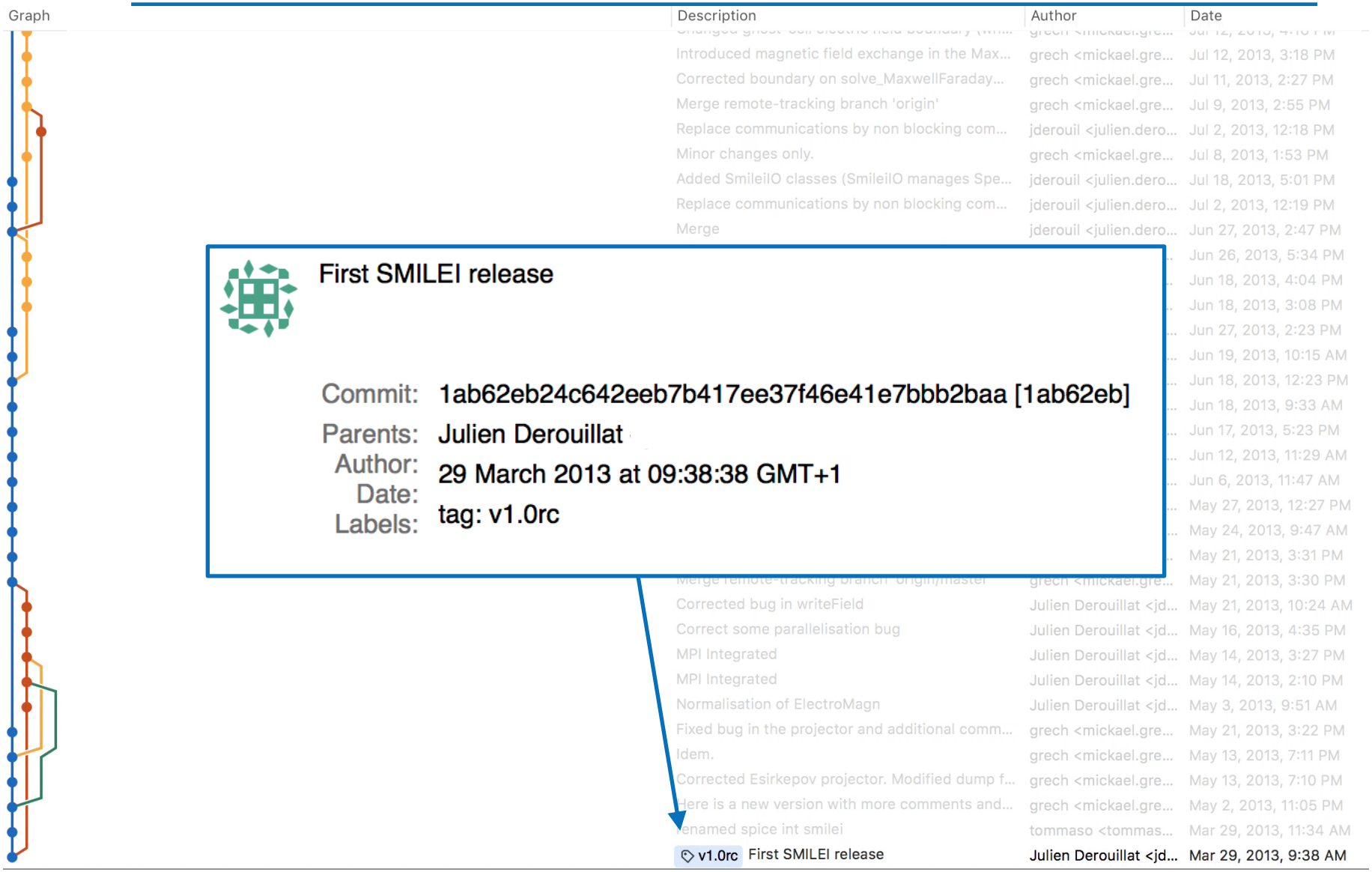
17 developers

Julien Derouillat	1058
Fred Perez	878
Tommaso Vinci	655
Arnaud Beck	635
Mickael Grech	278
Mathieu Lobet	275
Marco Chiaramello	72
Marie Fle	45
Haithem Kallala	30
Charles Ruyer	20
Mohamed Gaalich	10
Andrea Sgattoni	7
Jeremie Gaidamour	6
Anna Grassi	9
Jérémy Dargent	4
Nicolas Aunai	4
Anton Golovanov	2
Total commits	**3988**

50k lines

Language	files	comment	code
C++	108	5973	22192
Python	144	2498	13809
C/C++ Header	131	2872	9639
make	2	58	319
Bourne Shell	6	115	261
CSS	1	34	229
DOS Batch	1	2	227
HTML	5	49	160
YAML	2	10	60
Markdown	1	0	5
	401	**11611**	**46901**

version control system: Git



version control system: Git

Graph	Description	Author	Date
	Introduced magnetic field exchange in the Max...	grech <mickael.gre...	Jul 12, 2013, 3:18 PM
	Corrected boundary on solve_MaxwellFaraday...	grech <mickael.gre...	Jul 11, 2013, 2:27 PM
	Merge remote-tracking branch 'origin'	grech <mickael.gre...	Jul 9, 2013, 2:55 PM
	Replace communications by non blocking com...	jderouil <julien.dero...	Jul 2, 2013, 12:18 PM
	Minor changes only.	grech <mickael.gre...	Jul 8, 2013, 1:53 PM
	Added SmileiIO classes (SmileiIO manages Spe...	jderouil <julien.dero...	Jul 18, 2013, 5:01 PM
	Replace communications by non blocking com...	jderouil <julien.dero...	Jul 2, 2013, 12:19 PM
	Merge	jderouil <julien.dero...	Jun 27, 2013, 2:47 PM
	2D version of the code is implemented, it compi...	grech <mickael.gre...	Jun 26, 2013, 5:34 PM
	added bibtex to doc: edit the file in doc/smilei.bib	tommaso <tommas...	Jun 18, 2013, 4:04 PM
	changed utf8 char that prevented documentatio...	tommaso <tommas...	Jun 18, 2013, 3:08 PM
	Introduced v0 of 2D HDF5 storage in SmileiMPI_...	jderouil <julien.dero...	Jun 27, 2013, 2:23 PM
	Cartesian 2D solver integrated (sequential only)	jderouil <julien.dero...	Jun 19, 2013, 10:15 AM
	Modified to not manage directly communication...	jderouil <julien.dero...	Jun 18, 2013, 12:23 PM
	Introduced generic Cartesian XD exchangeParti...	jderouil <julien.dero...	Jun 18, 2013, 9:33 AM
	Introduced automatic splitting for 2D domain de...	jderouil <julien.dero...	Jun 17, 2013, 5:23 PM
	Modified Field constructors (add constructor sp...	jderouil <julien.dero...	Jun 12, 2013, 11:29 AM
	Maxwell solved on the whole local subdomain	jderouil <julien.dero...	Jun 6, 2013, 11:47 AM
	Test ECLIPSE environment	jderouil <julien.dero...	May 27, 2013, 12:27 PM
	Correct bug in exchangeField	Julien Derouillat <jd...	May 24, 2013, 9:47 AM
	Merge between MG & JD versions	grech <mickael.gre...	May 21, 2013, 3:31 PM
	Merge remote-tracking branch 'origin/master'	grech <mickael.gre...	May 21, 2013, 3:30 PM
	Corrected bug in writeField	Julien Derouillat <jd...	May 21, 2013, 10:24 AM
	Correct some parallelisation bug	Julien Derouillat <jd...	May 16, 2013, 4:35 PM
	MPI Integrated	Julien Derouillat <jd...	May 14, 2013, 3:27 PM
	MPI Integrated	Julien Derouillat <jd...	May 14, 2013, 2:10 PM
	Normalisation of ElectroMagn	Julien Derouillat <jd...	May 3, 2013, 9:51 AM
	Fixed bug in the projector and additional comm...	grech <mickael.gre...	May 21, 2013, 3:22 PM
	Idem.	grech <mickael.gre...	May 13, 2013, 7:11 PM
	Corrected Esirkepov projector. Modified dump f...	grech <mickael.gre...	May 13, 2013, 7:10 PM
	Here is a new version with more comments and...	grech <mickael.gre...	May 2, 2013, 11:05 PM
	renamed spice int smilei	tommaso <tommas...	Mar 29, 2013, 11:34 AM
	v1.0rc First SMILEI release	Julien Derouillat <jd...	Mar 29, 2013, 9:38 AM

two repositories

https://github.com/SmileiPIC/Smilei

GitHub

Public

online documentation
public issues

two repositories

https://github.com/SmileiPIC/Smilei

https://llrgit.in2p3.fr/smilei/smilei

Public

online documentation
public issues

Private

development branches
development issues

two repositories

https://github.com/SmileiPIC/Smilei

https://llrgit.in2p3.fr/smilei/smilei

GitHub

GitLab

Public

online documentation
public issues

SYNC

Private

development branches
development issues

get the sources

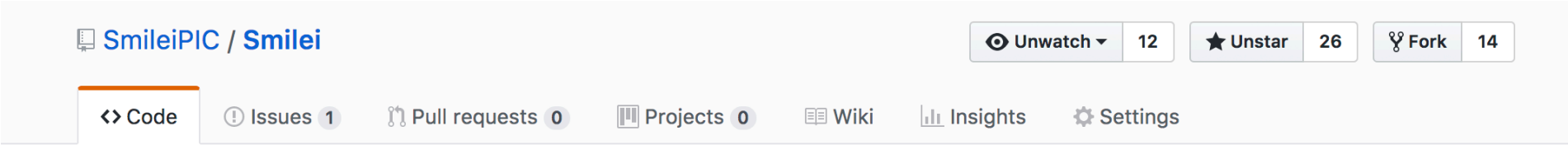
GitHub

```
git clone https://github.com/SmileiPIC/Smilei.git
```

get the sources

GitHub

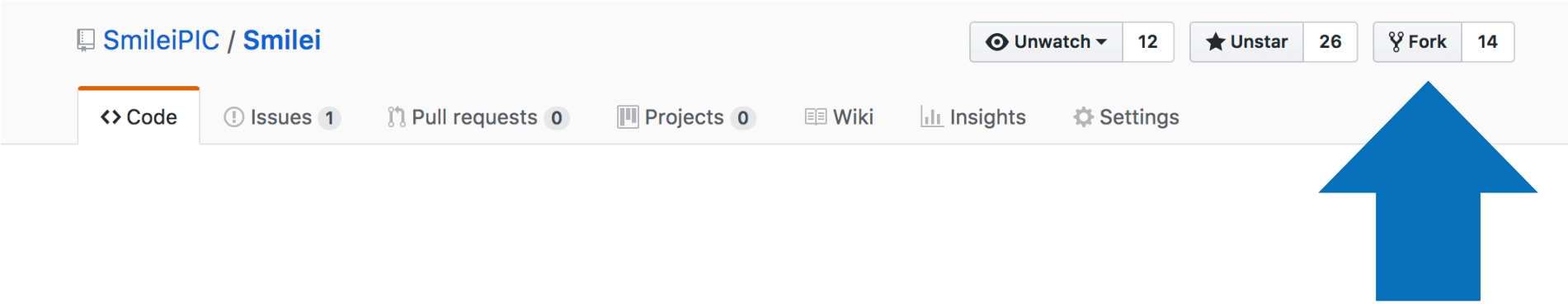
```
git clone https://github.com/SmileiPIC/Smilei.git
```



get the sources

GitHub

```
git clone https://github.com/SmileiPIC/Smilei.git
```



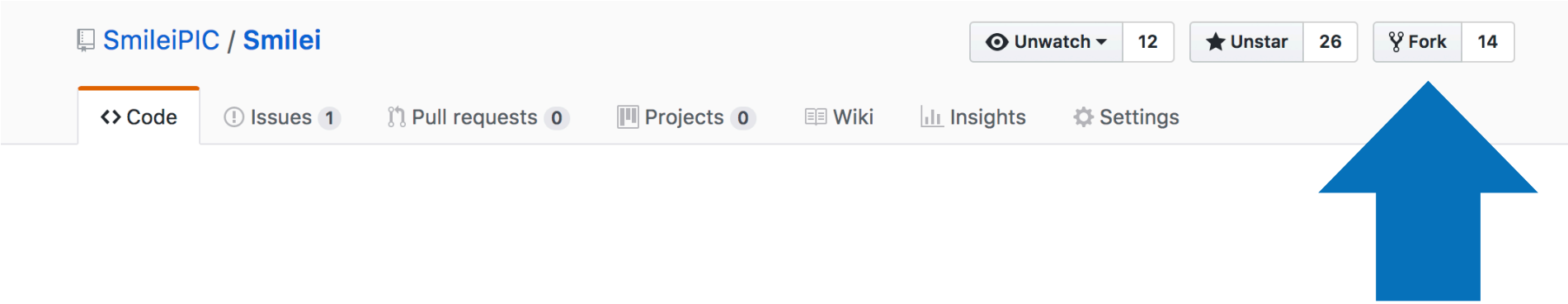

```
git clone https://github.com/username/Smilei.git
```

fork the code

get the sources

GitHub

```
git clone https://github.com/SmileiPIC/Smilei.git
```




```
git clone https://github.com/username/Smilei.git
```

get the documentation

Smilei

Licence
Synopsis
Highlights
Releases

Units
PIC algorithms

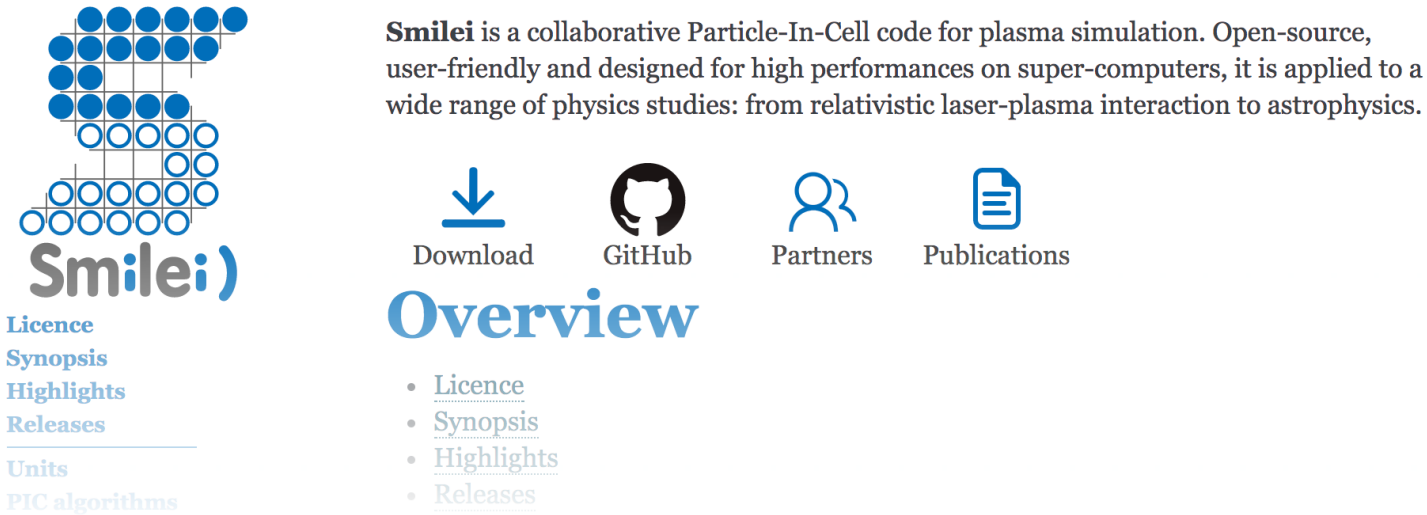
Smilei is a collaborative Particle-In-Cell code for plasma simulation. Open-source, user-friendly and designed for high performances on super-computers, it is applied to a wide range of physics studies: from relativistic laser-plasma interaction to astrophysics.

Download GitHub Partners Publications

Overview

- Licence
- Synopsis
- Highlights
- Releases

get the documentation

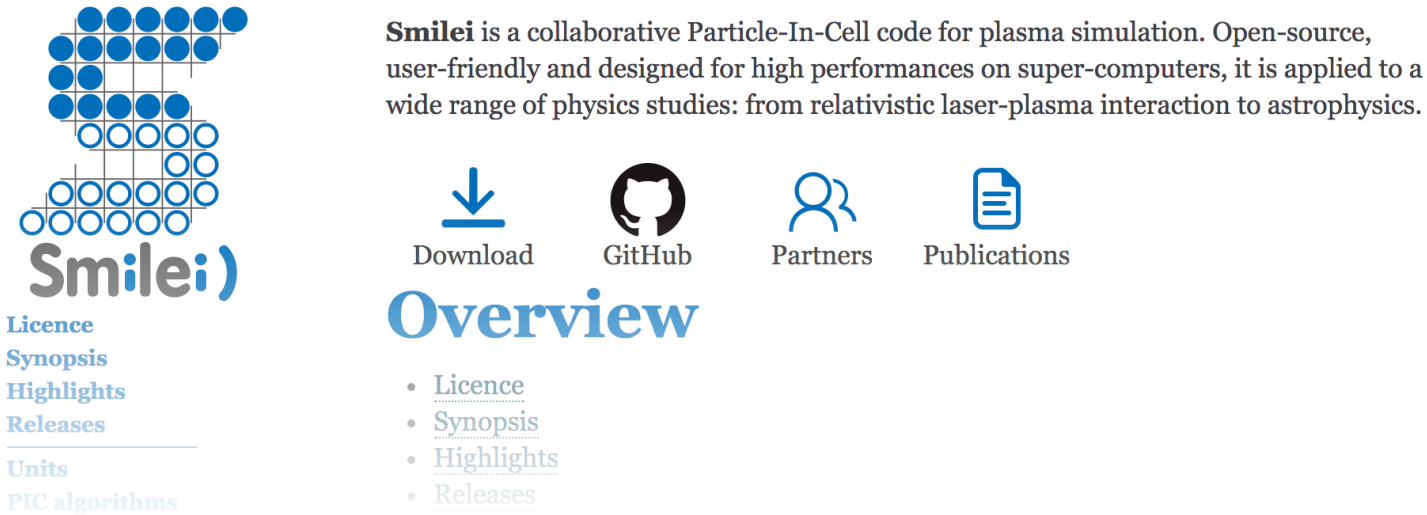


online:

smileipic.github.io/Smilei

www.maisondelasimulation.fr/smilei

get the documentation



online:

smileipic.github.io/Smilei

www.maisondelasimulation.fr/smilei

on your machine:

```
make doc
```

restructured text

```
Write a namelist
----------------

Before you run :program:`Smilei`, you need a *namelist* (an input file). The namelist
is written in the *python* language. It is thus recommended to know the basics of *python*.

We suggest you copy one existing namelist from the folder *benchmarks*.
All namelists have the extension ``.py``.


----

General rules
^^^^^^^^^^^^^

* :program:`Smilei` requires a few *blocks* to be defined, such as::

    Main(
        # ...
        timestep = 0.01,          # defines the timestep value
        grid_length = [10., 20.], # defines the 2D box dimensions
        # ...
    )

  Outside blocks, you can calculate anything you require.
  Inside a block, you must only define variables for :program:`Smilei`.
```

Write a namelist

Before you run **Smilei**, you need a *namelist* (an input file). The namelist is written in the *python* language. It is thus recommended to know the basics of *python*.

We suggest you copy one existing namelist from the folder *benchmarks*. All namelists have the extension `.py`.

SPHINX

General rules

- **Smilei** requires a few *blocks* to be defined, such as:

```
Main(
    # ...
    timestep = 0.01,          # defines the timestep value
    sim_length = [10., 20.], # defines the 2D box dimensions
    # ...
)
```

Outside blocks, you can calculate anything you require. Inside a block, you must only define variables for **Smilei**.

improve docs

restructured text

Write a namelist

Before you run **Smilei**, you need a *namelist* (an input file). The namelist is written in the *python* language. It is thus recommended to know the basics of *python*.

We suggest you copy one existing namelist from the folder *benchmarks*. All namelists have the extension `.py`.

SPHINX

General rules

- **Smilei** requires a few *blocks* to be defined, such as:

```
Main(
    # ...
    timestep = 0.01,          # defines the timestep value
    sim_length = [10., 20.], # defines the 2D box dimensions
    # ...
)
```

Outside blocks, you can calculate anything you require. Inside a block, you must only define variables for **Smilei**.

compile

few dependencies: hdf5, python, numpy

Makefile
THE ORIGINAL BUILD TOOL

compile

few dependencies: hdf5, python, numpy

Makefile
THE ORIGINAL BUILD TOOL

for non-standard machines:

`scripts/CompileTools/machine/poincare`

compile

few dependencies: hdf5, python, numpy

Makefile
THE ORIGINAL BUILD TOOL

for non-standard machines:

`scripts/CompileTools/machine/poincare`

```
# module load intel/15.0.0 intelmpi/5.0.1 hdf5/1.8.16_intel_intelmpi_mt …
# unset LD_PRELOAD
# export PYTHONHOME=/gpfslocal/pub/python/anaconda/Anaconda-2.1.0

LDFLAGS += -lgpfs -lz -L/gpfslocal/pub/python/anaconda/Anaconda-2.1.0/lib
CXXFLAGS += -xAVX
```

compile

few dependencies: hdf5, python, numpy

Makefile
THE ORIGINAL BUILD TOOL

for non-standard machines:

`scripts/CompileTools/machine/poincare`

```
# module load intel/15.0.0 intelmpi/5.0.1 hdf5/1.8.16_intel_intelmpi_mt …
# unset LD_PRELOAD
# export PYTHONHOME=/gpfslocal/pub/python/anaconda/Anaconda-2.1.0

LDFLAGS += -lgpfs -lz -L/gpfslocal/pub/python/anaconda/Anaconda-2.1.0/lib
CXXFLAGS += -xAVX
```

```
make machine=poincare
make machine=poincare help
```

test new machines

compile

few dependencies: hdf5, python, numpy

Makefile
THE ORIGINAL BUILD TOOL

for non-standard machines:

`scripts/CompileTools/machine/poincare`

```
# module load intel/15.0.0 intelmpi/5.0.1 hdf5/1.8.16_intel_intelmpi_mt …
# unset LD_PRELOAD
# export PYTHONHOME=/gpfslocal/pub/python/anaconda/Anaconda-2.1.0

LDFLAGS += -lgpfs -lz -L/gpfslocal/pub/python/anaconda/Anaconda-2.1.0/lib
CXXFLAGS += -xAVX
```

```
make machine=poincare
make machine=poincare help
```

where to start

95 test cases

where to start

95 test cases

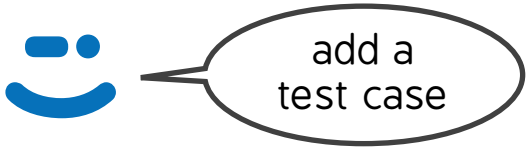
obviously located in the **`benchmarks`** directory ....

Geometry

Physical meaning

`tst2d_2_radiation_pressure_acc.py`

progressive number



where to start

95 test cases

obviously located in the **`benchmarks`** directory ....

Geometry

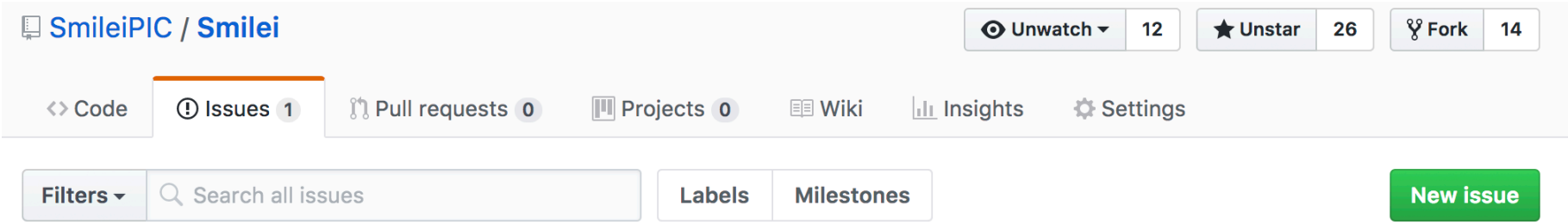
Physical meaning

`tst2d_2_radiation_pressure_acc.py`

progressive number

troubles?

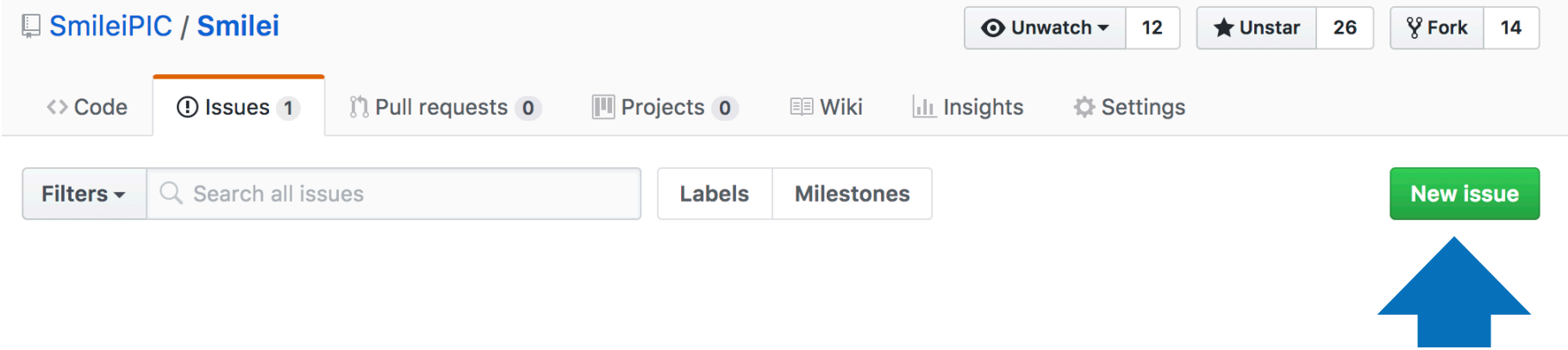
GitHub



report bugs

troubles?

GitHub

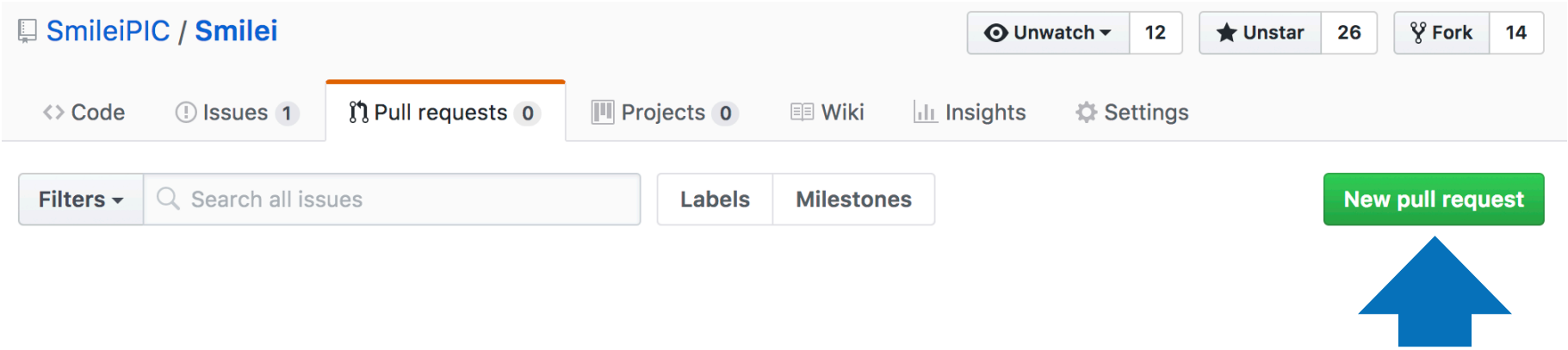


integration workflow

1. modify your local fork on GitHub

integration workflow

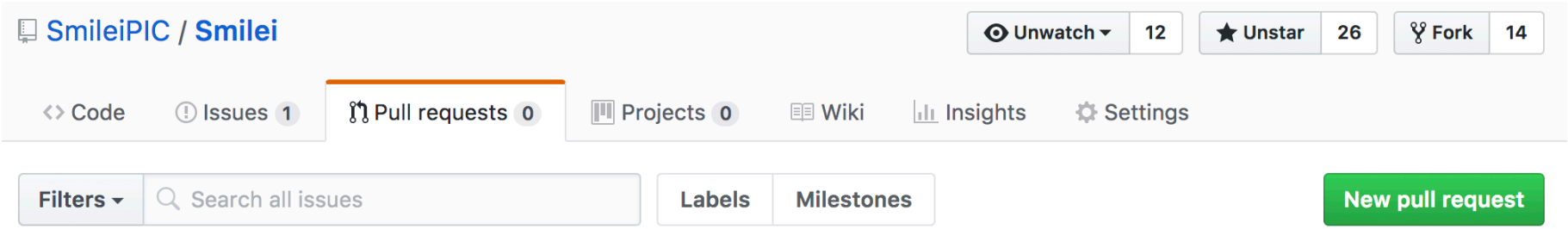
1. modify your local fork on GitHub
2. create a pull request



integration workflow

1. modify your local fork on GitHub

2. create a pull request

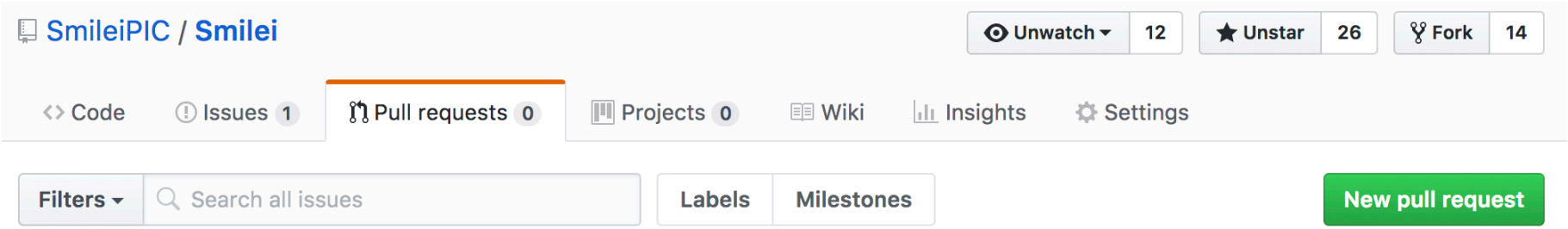


3. we test on GitLab

integration workflow

1. modify your local fork on GitHub

2. create a pull request



3. we test on GitLab

OK

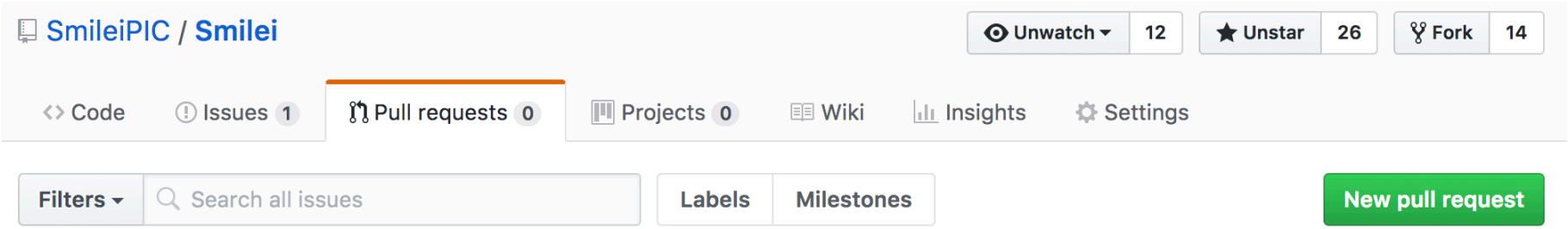
4. feature / fix is integrated on GitHub

develop
or fix

integration workflow

1. modify your local fork on GitHub

2. create a pull request



3. we test on GitLab

OK

4. feature / fix is integrated on GitHub

continuous integration

continuous integration

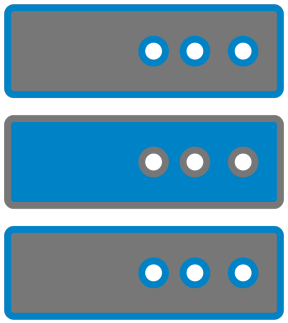
continuous integration

Test with 4 MPI and 4 OpenMP:

- compilation
- physical tests
- code consistency

continuous integration

GitLab



Test with 4 MPI and 4 OpenMP:

- compilation
- physical tests
- code consistency

passed **Pipeline #1495**

Fix bug compilation in debug mode

5 jobs from develop in 23 minutes 9 seconds (queued for 2 seconds)

continuous integration

GitLab

Test with 4 MPI and 4 OpenMP:

- compilation
- physical tests
- code consistency

passed **Pipeline #1495**

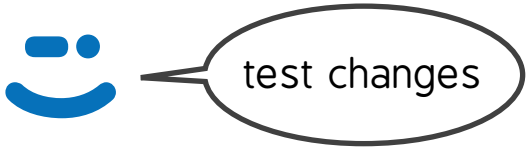
Fix bug compilation in debug mode

5 jobs from `develop` in 23 minutes 9 seconds (queued for 2 seconds)

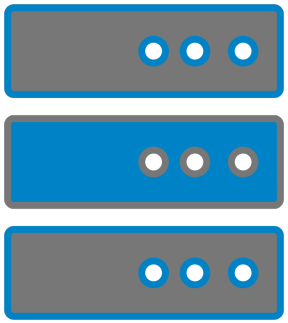
failed **Pipeline #1491**

correction of 3D4Order interpolator

6 jobs from `develop` in 16 minutes 32 seconds (queued for 3 seconds)



continuous integration



Test with 4 MPI and 4 OpenMP:

- compilation
- physical tests
- code consistency

passed **Pipeline #1495**

Fix bug compilation in debug mode

5 jobs from `develop` in 23 minutes 9 seconds (queued for 2 seconds)

failed **Pipeline #1491**

correction of 3D4Order interpolator

6 jobs from `develop` in 16 minutes 32 seconds (queued for 3 seconds)

validation/validation.py

validation/validation.py

1. benchmark file
   benchmarks/tst1d_0_em_propagation.py

validation/validation.py

1. benchmark file
   benchmarks/tst1d_0_em_propagation.py

2. python script
   validation/validate_tst1d_0_em_propagation.py

validation/validation.py

1. benchmark file
   benchmarks/tst1d_0_em_propagation.py

2. python script
   validation/validate_tst1d_0_em_propagation.py

```python
import os, re, numpy as np, h5py
from Smilei import *

S = Smilei(".", verbose=False)

# FIELD DIAG
Validate("List of fields in Field0", S.Field.Field0().getFields() )
```

[...]

```python
max_ubal = np.max( np.abs(S.Scalar.Ubal().getData()) )
Validate("Max Ubal is below 2%", max_ubal<0.02 )
```

validation/validation.py

1. benchmark file
   benchmarks/tst1d_0_em_propagation.py

2. python script
   validation/validate_tst1d_0_em_propagation.py

```python
import os, re, numpy as np, h5py
from Smilei import *

S = Smilei(".", verbose=False)

# FIELD DIAG
Validate("List of fields in Field0", S.Field.Field0().getFields() )
```

[...]

```python
max_ubal = np.max( np.abs(S.Scalar.Ubal().getData()) )
Validate("Max Ubal is below 2%", max_ubal<0.02 )
```

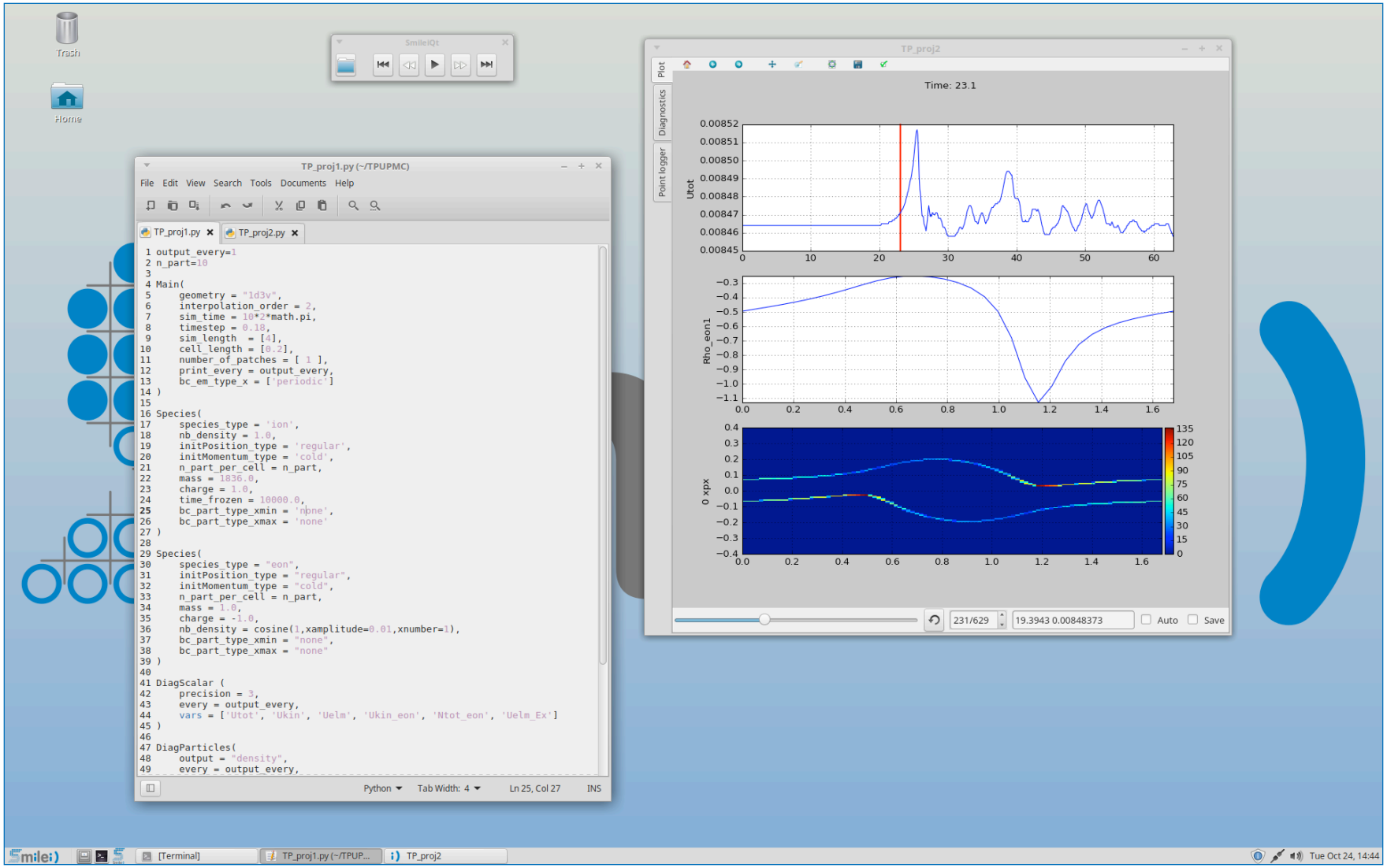
3. reference file
   validation/references/tst1d_0_em_propagation.py.txt

not only HPC

3rd year of practicals @UPMC https://github.com/SmileiPIC/TPUPMC

not only HPC

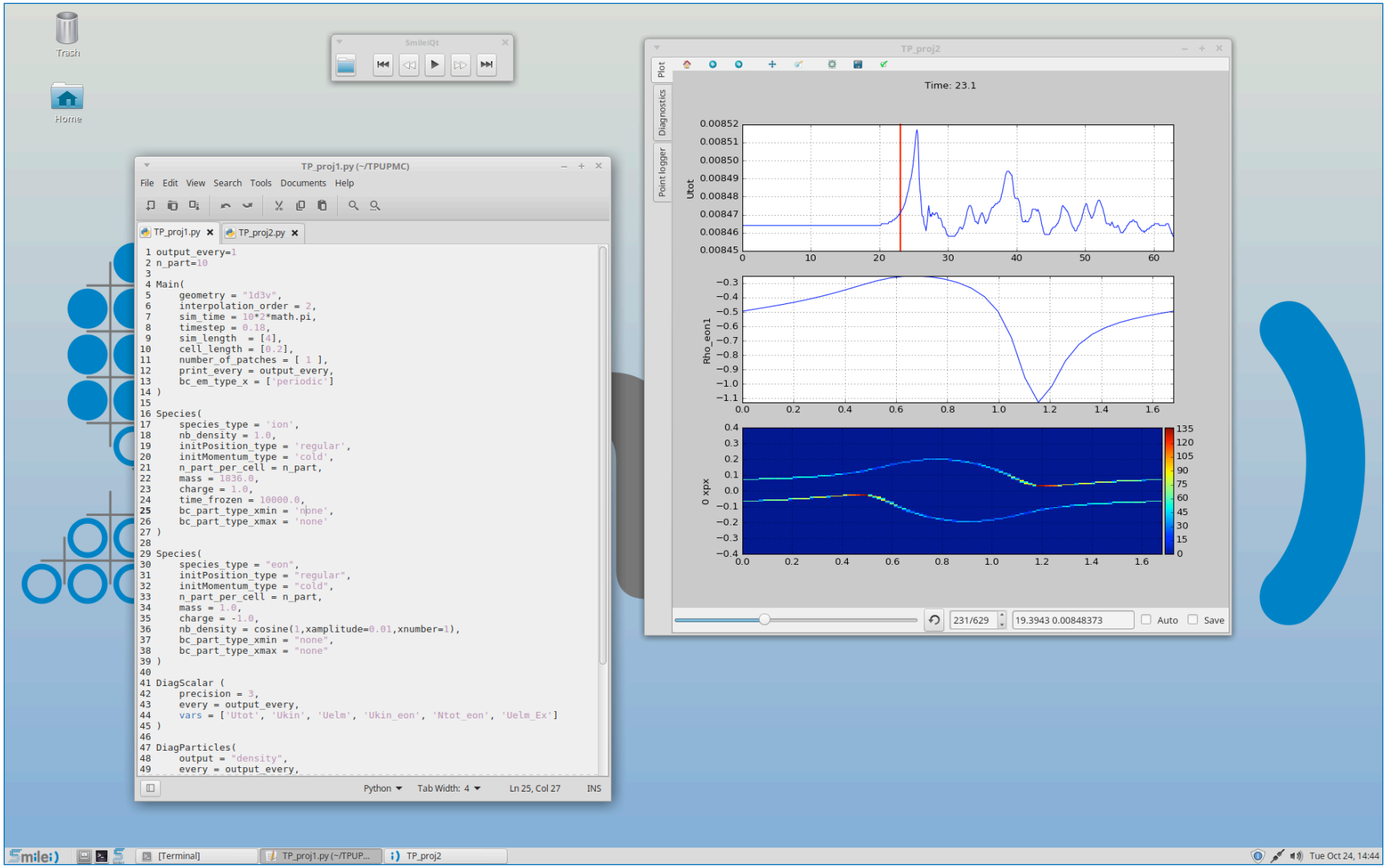
3rd year of practicals @UPMC https://github.com/SmileiPIC/TPUPMC



use with students

not only HPC

3rd year of practicals @UPMC https://github.com/SmileiPIC/TPUPMC



summary

summary

- get Smilei on new machines / architectures
- provide / expand docs
- report problems
- provide benchmarks / new physical test cases
- let the students play
- code a new feature / fix an issue

be happi)
keep Smilei)

summary

- get Smilei on new machines / architectures
- provide / expand docs
- report problems
- provide benchmarks / new physical test cases
- let the students play
- code a new feature / fix an issue